

Nbr.6 Vol.6

Price \$1.50

May 1986

MILATARI MEETING SCHEDULE - MAY 1986

Ambruster School - Saturday, May 17, 1986

Begins at 2:15 P.M.

2:15 PM BASIC Class - Room 107B
Instructor - Steve Armstrong

2:15 PM ST SIG meeting - Room 107A
SIG leader - Gary Nolan

3:30 PM BUSINESS MEETING - Gym
Election of new officers:

Report from the nominating committee;

For President.....Ron Friedel
Vice President...Steve Tupper
Secretary.....Steve Armsrtong
Treasurer.....Carl Mielcarek

The floor will entertain additional
nominations from the floor.

4:00 PM Demonstrations - GAMES!!!!!!!!!!!!!!!!!!!!!!

Carl Mielcarek has put together another
exciting program this month. He has rounded
up four of the hottest new games for 8 and 16
bit Ataris. We will set up a demo center in
each corner of the gym. Look forward to fast
action, great graphics and lots of fun!

COMING ATTRACTIONS

June - Hybrid Art's MIDI interface for 8 bit ATARIS

July - MILATARI's annual summer swap fest and computer garage sale

August - 3rd annual MILATARI family picnic



THE FUZZY NELAN REVIEW

BY GARY NELAN



IN WITH THE OLD, OUT WITH THE NEW

Somehow that doesn't sound right. But on with it! May is election month at Milatari and the nomination committee has selected a slate for your consideration. So lets all show up at the May meeting and vote for the officers who will have the task of running this outfit for another year. You'll find the slate published elsewhere in the newsletter, but you can nominate anyone you feel would do a good job at next months meeting if you want. That old saying still applies, "Get out and VOTE".

LISTEN HELEN, THEY'RE PLAYING OUR SONG

From Activision comes The Music Studio, a music program that lets you edit and arrange any style of music easily and completely. The music prints out in standard sheet music form. You can design your own instruments or use the built-in MIDI capability to interface with electronic keyboards. It will run on the 800, XL, XE and ST computers. List price is around \$60. If any of you have the Hybrid Arts MIDITRACK II or III Interface, you can upgrade your software. Faris Compu-Mates is offering the upgrades along with several other offers. There are several 130XE/800XL memory upgrades and a voice module called Quiggy that will serve as the base of a voice recognition system package. This package consists of a dictionary disk, editor disk, programming aids in all languages and the module itself. All this for \$80. They are offering special pricing to Milatari members on all their products. If you're interested and can't wait till the meeting give them a call at (213) 271-7410. If you're interested in a MIDI interface for your 8-bit machine you'll want to be at the June meeting. We will be having a demo of the Hybrid Arts unit by Bob Derrmarais of Cascio Music Co. in New Berlin. Should be very interesting and entertaining.

SURE, NOW THEY TELL US

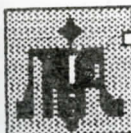
(BUT WHY ARE WE ALWAYS THE LAST TO KNOW?)

I've read in several trade papers that Atari will be running a special promo through the end of May. If you buy an 520ST system during the month you can buy an additional disk drive for only \$70. I haven't seen any ads in any of the mags or papers, but keep your eyes open.

You've probably heard about the IBM emulator board that Atari is planning to release, well they also have made noise about putting out a board for the IBM PC that will allow it to run ST programs. The Atari to IBM board will run about \$200.

At last months ST SIG meeting I mentioned that Atari was working on raising the res on its planned 32 bit machine to 1024 X 1024. Now I hear that that's in the low res mode, high res will be 2048 X 2048. And DJ plans to have a board available for the 520/1040 ST's with a 68020 on board that will allow these units to run UNIX, as per the new systems.

This sounds good for US. But if you're a dealer all this "good news" came before they threw the bucket of ice water on you. "Daddy Jack" at a press conference in New York announced that, "I don't serve dealers. I serve end users." Which means that when "NEW" products come out, the old ones are moved to the mass merchants. With DJ at his side an aide told the press, "When new products come, the 1040 will be in Toys R Us." Now this should make all the business people who might be thinking about buying an ST happy. I mean hey, they won't have to drive very far for support. After all,



NEWS FROM THE NORTH

(YOU SURE SGT. PRESTON STARTED THIS WAY)

In looking through the Phoenix, a newsletter put out by the Toronto Atari user group a couple of interesting things caught my eye. It seems that Atari (Canada) puts out a newsletter that it sends to the user groups. (My what a novel idea. Hint, Hint "Daddy J") Anyway, in it they say that there will be an 80 col card for the XE that will have its own parallel printer port. Supposed to be out this spring. And something that I'm looking forward to, the IBM emulator will be out this summer. (Maybe it'll be at CES) And supposedly there are three (3) versions of the 520ST boards and each requires a different mod when installing the TOS ROM chips. This could be some members had trouble getting their computers to work after installing the TOS chips. What ever happened to Atari's "new attitude" in sharing technical info with the users groups ie: their customer base?? AKA, the "End User". What say, DJ???

If any of you ST people are having trouble with noise interference from your ST's we will publish a fix that should cure the problem. It involves soldering a 1000uf capacitor onto the motherboard.

Atari is coming out with a 3 1/2" drive for you 8-bit people. The XE351 will have 327K of formatted storage and should be available around mid-year.

SOLAR WHAT???

From Tim Hunkler and his company Solar Powered Software come three programs for the ST's. The first is SOLADISK v1.2 and is a ramdisk program written in assembly language and allows you to adjust the file size and the ramdisk size. Ramdisk sizes up to one meg are supported and can be installed automatically during boot-up.

Second is SOLASAVE v1.2, a CRT screen saver utility that can also be installed at boot-up. This is an "attract" program for the ST's just like the 8-bit systems have. It will help prevent burn-in if the computer is left unattended for an extended period of time. It is sold as a companion program to SOLADISK and it's price includes both programs.

Last comes SOLAPAK v1.0, which combines #1 & #2 with a print spooler. The PS can be installed as a desk utility and is then instantly available for use. Up to 8 files can be queued for printing and each file can have separate print qualities. The program has 32 user redefinable print buttons as well as 4 redefinable preset buttons. It can be configured to support almost any parallel printer and the print speed can be adjusted to handle printer rated up to 300cps.

Interested? Pricing is like this.

SOLADISK \$15 + \$1 handling

SOLASAVE \$18 + \$1 handling

SOLAPAK \$24 + \$1 handling

These are special User Group prices so mention that you are a member of Milatari and read about it in this newsletter. You can order send your check or money order (sorry, no plastic) to:

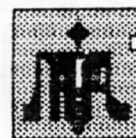
Solar Power Software

1807 N. Evergreen

Chandler, AZ 85224

If you want more info before ordering, see me at the meeting and look over the fact sheet.

**MILATARI BBS 414-781-5710 24 Hours
300 Baud**



SOMEBODY NOTICED!!!!!!! (OR, HEY! THAT'S US IN ANTIC)

In the May issue of Antic, the magazine gave out its first Antic Awards For Atari Achievements. They gave "Daddy Jack" the Atarian of the Year award, OSS received a Lifetime Contribution Award (GOOD CHOICE) and several product awards other types were given. The Outstanding User Group award went to the Eugene ACE group from Eugene, OR. Again another good choice. Milatari was mentioned along with several other groups in the article. Pick it up and look through the awards section, very interesting. The one award they did not give out was one for an Atari publication. And if an independent firm had given the awards I'm sure Antic would have won. Thanks for all the help guys, on both an individual and club level.

GINMEE-GINMEE-GINMEE DEM THINGS

The rumor mill says that "Daddy J" is just waiting for the OEM price on CD ROM units to reach a point where he can bring one out to the consumer (that's us'ns kids) for around \$600. Gee, I paid \$525 for my first 810 disk drive, and that was a GOOD price at the time.

Now the latest tale has our man in Sunnyvale looking for a source of Laser printers that he can sell for \$1500/\$2000. We're talking about a full blown, fast shootin' machine with most of the goodies.

Commodore may push "DJ" into bringing out the IBM add-on in a hurry. Seems the water's breaking over the bow and Commodore is going to bring its IBM clone machine in from Europe to boost sales. With a \$1200 price tag and a name most people have heard of, it could be a good selling item for them.

Hmmm, lets see. A 1040 with 2Meg of RAM, A CD ROM (maybe a write once type?) and a laser printer running IBM software for under \$3500. Gimme dem things, indeed!

SSI has announced that they will be bringing out an ST version of their popular word processor, Word Perfect, later this year. I've worked with this program on an MS-DOS machine and like it a lot. It will be a welcomed addition to the ST's software base.

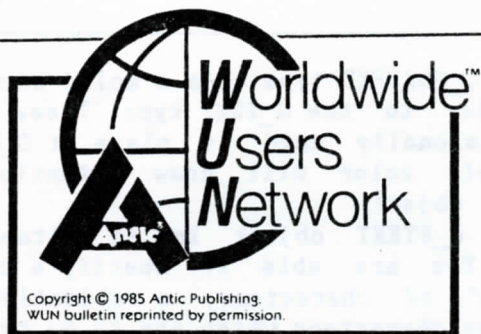
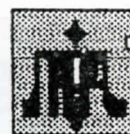
Migraph has released Easy-Draw, which they call a "professional" drawing program for the ST's. Unlike "painting" programs which are bit-map based, E-D is object oriented and supplies the user with a set of 11 tools like squares, circles and lines which allows them to copy, stretch and size objects as they are drawn. E-D is the first drawing program to take advantage of the GEM interface, they claim. And rather than just doing a screen dump when printing your creation, E-D drives most output devices at their highest resolution. The program will run on both mono and color monitors and currently supports the Epson FX-80 printer, with additional drivers for laser and color dot matrix printers coming soon.

TID - BITS

The April 28th issue of Infoworld had a short announcement about the expandable (w/card slots) Macintosh which has been code named the Milwaukee. Would I lie to you?

It's been reported that, mah man, Nolan Bushnell (Atari founder) and Steve Wozniak (one of Apple's founders) are getting together for a joint project. Are you ready for a Petster with expansion slots??

Habba/Arrays Inc has announced its 10M hard disk for the ST's. Selling for \$700 the unit has a transfer rate of 5M bits per second and is a self contained unit, which means it has its own power supply and interface cable.



Copyright © 1985 Antic Publishing.
WUN bulletin reprinted by permission.

Permission to reprint or excerpt is granted only if the following line appears at the top of the article:

ANTIC PUBLISHING INC., COPYRIGHT 1985.
REPRINTED BY PERMISSION.

Welcome to the fourth installment of ST PRO GEM. We are about to delve into the mysteries of GEM resource structure, and then use this knowledge to create some useful utilities for handling dialogs.

The first and largest listing contains a C image of a sample resource file. To create this listing, I used the GEM Resource Construction Set to create a dummy resource with three dialogs including examples of all object types, then enabled the C output option and saved the resource. If you have access to a copy of RCS, I suggest that you create your own listing in order to get a feel for the results. Then, using either listing as a roadmap to the resource, you can follow along as we enter...

A MAZE OF TWISTY LITTLE PASSAGES.

While a GEM resource is loaded as a block of binary information, it is actually composed of a number of different data structures. These structures are linked together in a rather tangled hierarchy. Our first job is to map this linkage system.

The topmost structure in a resource file is the resource header. This is an array of words containing the size and offset within the resource of the other structures which follow. This information is used by GEM during the resource load process, and you should never need to access it. (The resource header does not appear in the C output file; it is generated by the RSCREATE utility if the C file is used to recreate the resource.)

The next structure of interest is the tree index. This is an array of long pointers, each which addresses the beginning of an object tree. Again, you wouldn't normally access this structure directly. The GEM `rsrc_gaddr` call uses it when finding trees' addresses. This structure is called "rs_trindex" in the C output,

If you look at the contents of `rs_trindex` you will notice that the values are integers, instead of the pointers I described. What has happened is that RCS has converted the pointers to indices into the object array. (If you actually used the C file to recreate the resource file, then the pointers would be regenerated by RSCREATE.)

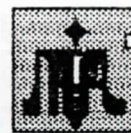
Now you can follow the link from `rs_trindex` to the objects stored in `rs_object`. Take (for instance) the second entry in `rs_trindex` and count down that many lines in `rs_object`. The following line (object) should start with a -1. This indicates that it is the root object of a tree. The following objects down to the next root belong to that tree. We'll pass over the details of inter-object linkage for now, leaving it for a later column.

There are a number of different fields in an object, but right now we'll concentrate on two of them: `OB_TYPE` and `OB_SPEC`. The `OB_TYPE` is the field which contains mnemonics like `G_STRING` and `G_BOX` indicating the type of the object. The `OB_SPEC` is the only field in each object which is a LONG - you can tell it by the L after the number.

What's in `OB_SPEC` depends on the object type, so we need to talk about what kinds of objects are available, what you might use them for, and finally how they use the `OB_SPEC` field.

The box type objects are `G_BOX`, `G_IBOX`, and `G_BOXCHAR`. A `G_BOX` is an opaque rectangle, with an optional border. It's used to create a solid patch of color or pattern on which to place other objects. For instance, the background of a dialog is a `G_BOX`.

A `G_IBOX` is a hollow box which has only a border. (If the border has no thickness, then the box is "invisible", hence the name.) The favorite use for `IBOX`s is to hold radio buttons. There is also one neat trick you can play with an `IBOX`. If you have more than one object (say an image and a string) which you would like to have selected all at once, you can insert them in a dialog, then cover them with an `IBOX`. Since the box is transparent, they will show through. If you now make the box selectable, clicking on it will highlight the whole area at once!



The G_BOXCHAR is just like a G_BOX, except that a single character is drawn in its center. They are mostly used as "control points": the FULLER, CLOSER, SIZER, and arrows in GEM windows are BOXCHARs, as are the components of the color selection gadgets in the RCS.

The OB_SPEC for box type objects is a packed bit array. Its various fields contain the background color and pattern, the border thickness and color, and the optional character and its color.

The string type objects are G_STRING, G_BUTTON, and G_TITLE. G_STRINGs (in addition to being a bad pun) are for setting up static explanatory text within dialogs. The characters are always written in the "system font": full size, black, with no special effects.

We have already discussed many of the uses of G_BUTTONs. They add a border around the text. The thickness of a G_BUTTON's border is determined by what flags are set for the object. All buttons start out with a border thickness of one pixel. One pixel is added if the EXIT attribute is set, and one more is added if the DEFAULT attribute is set.

The G_TITLE type is a specially formatted text string used only in the title bar of menus. This type is needed to make sure that the menus redraw correctly. The Resource Construction Set automatically handles inserting G_TITLES, so you will seldom use them directly.

In a resource, the OB_SPEC for all string objects is a long pointer to a null terminated ASCII string. The string data in the C file is shown in the BYTE array rs_strings. Again you will notice that the OB_SPECS in the C file have been converted to indices into rs_string. To find the string which matches the object, take the value of OB_SPEC and count down that many lines in rs_strings. The next line is the correct string.

The formatted text object types are G_TEXT, G_BOXTEXT, G_FTEXT, and G_FBOXTEXT. G_TEXTs are a lot like strings, except that you can specify a color, different sizes, and a positioning rule for the text. Since they require more memory than G_STRINGs, G_TEXTs should be used sparingly to draw attention to important information within a dialog. G_TEXTs are also useful for automatic centering of dialog text which is changed at run-time. I will describe this technique in detail later on.

The G_BOXTEXT type adds a solid background and border to the G_TEXT type. These objects are occasionally used in place of G_BUTTONs when their color will draw attention to an important object.

The G_FTEXT object is an editable text field. You are able to specify a constant "template" of characters, a validation field for those characters which are to be typed in, and an initial value for the input characters. You may also select color, size, and positioning rule for G_FTEXTs. We'll discuss text editing at length below.

The G_FBOXTEXT object, as you might suspect, is the same as G_FTEXT with the addition of background and border. This type is seldom used: the extra appearance details distract attention from the text being edited. The OB_SPEC for a formatted text object is a pointer to yet another type of structure: a TEDINFO. In the C file, you will find these in rs_tedinfo. Take the OB_SPEC value from each text type object and count down that many entries in rs_tedinfo, finding the matching TEDINFO on the next line. Each contains pointers to ASCII strings for the template validation, and initialization. You can find these strings in rs_strings, just as above.

There are also fields for the optional background and border details, and for the length of the template and text. As we will see when discussing editing, the most important TEDINFO fields are the TE_PTEXT pointer to initialized text and the TE_TXTLEN field which gives its length.

The G_IMAGE object type is the only one of its kind. A G_IMAGE is a monochrome bit image. For examples, see the images within the various GEM alert boxes. Note that monochrome does not necessarily mean black. The image may be any color, but all parts of it are the SAME color. G_IMAGES are used as visual cues in dialogs. They are seldom used as selectable items because their entire rectangle is inverted when they are clicked. This effect is seldom visually pleasing, particularly if the image is colored.

G_IMAGE objects have an OB_SPEC which is a pointer to a further structure type: the BITBLK. By now, you should guess that you will find it in the C file in the array rs_bitblk. The BITBLK contains fields describing the height and width of the image in pixels, its color, and it also contains a long pointer to the actual bits which make up the image. In the C file, the images are encoded as hexadecimal words and stored in arrays named IMAGO, IMAG1, and so on.



out it adds a mask array which selects what portions of the image will be drawn, as well as an explanatory text field. A G_ICON may also specify different colors for its "foreground" pixels (the ones that are normally black), and its "background" pixels (which are normally white).

The pictures which you see in Desktop windows are G_ICONS, and so are the disks and trashcan on the desktop surface. With the latter you will notice the effects of the mask. The desktop shows through right up to the edge of the G_ICON, and only the icon itself (not a rectangle) is inverted when a disk is selected.

The OB_SPEC of an icon points to another structure called an ICONBLK. It is shown in the C file as rs_iconblk. The ICONBLK contains long pointers to its foreground bit array, to the mask bit array, and to the ASCII string of explanatory text. It also has the foreground and background colors as well as the location of the text area from the upper left of the icon. The most common use of G_ICONS and ICONBLKs is not in dialogs, instead they are used frequently in trees which are built at run-time, such as Desktop windows. In a future article, we will return to a discussion of building such "on-the-fly" trees with G_ICONS.

Now, let's recap the hierarchy of resource structures: The highest level structures are the resource header, and then the tree index. The tree index points to the beginning of each object tree. The objects making up the tree are of several types, and depending on that type, they may contain pointers to ASCII strings, or to TEDINFO, ICONBLK, or BITBLK structures. TEDINFOs contain further pointers to strings; BITBLKs have pointers to bit images; and ICONBLKs have both.

PUTTING IT TO WORK. The most common situations requiring you to understand resource structures involve the use of text and editable text objects in dialogs. We'll look at two such techniques. Often an application requires two or more dialogs which are very similar except for one or two title lines. In this circumstance, you can save a good deal of resource space by building only one dialog, and changing the title at run time.

It is easy to go wrong with this practice, however, because the obvious tactic of using a G_STRING and writing over its text at run time can go wrong. The first problem is that you must know in advance the longest title to be used, and put a string that long into the resource. If you don't you will damage other

G_STRING is always drawn at the same place in a dialog. If the length of the title changes from time to time, the dialog will have an unbalanced and sloppy appearance.

A better way to do this is to exploit the G_TEXT object type, and the TEDINFO structure. The set_text() routine in the download shows how. The parameters provided are the tree address, the object number, and the 32-bit address of the string to be substituted. For this to work, the object referenced should be defined as a G_TEXT type object. Additionally, the Centered text type should be chosen, and the object should have been "stretched" so that it fills the dialog box from side to side.

In the code, the first action is to get the OB_SPEC from the object which was referenced. Since we know that the object is a G_TEXT, the OB_SPEC must point to a TEDINFO. We need to change two fields in the TEDINFO. The TE_PTEXT field is the pointer to the actual string to be displayed; we replace it with the address of our new string. The TE_TXTLEN field is loaded with the new string's length. Since the Centered attribute was specified for the object, changing the TE_TXTLEN will cause the string to be correctly positioned in the middle of the dialog!

Editing text also requires working with the TEDINFO structure. One way of doing this is shown in the download. The object to be used (EDITOBJ) is assumed to be a G_FTEXT or G_FBOXTEXT. Since we will replace the initialized text at run time, that field may be left empty when building the object in the RCS.

The basic trick of this code is to point the TEDINFO's TE_PTEXT at a string which is defined in your code's local stack. The advantages of this technique are that you save resource space, save static data by putting the string in reusable stack memory, and automatically create a scratch string which may be discarded if the dialog is cancelled.

The text string shown is arbitrarily 41 characters long. You should give yours a length equal to the number of blanks in the object's template field plus one. Note that the code is shown as a segment, rather than a subroutine. This is required because the text string must be allocated within the context of dialog handling routine itself, rather than a routine which it calls!

After the tree address is found, the code proceeds to find the TEDINFO and modify its TE_PTEXT as described above. However, the length which is inserted into TE_TXTLEN must be the maximum string length, including the null!



MILATAPI NEWSLETTER



PAGE 9

```
ICONBLK rs_iconblk[] = {
/* First pointer is mask */
/* Second is data, third */
/* is to title string */
1L, 2L, 10L, 4096,0,0, 0,0,48,24, 9,24,30,8,
3L, 4L, 17L, 4864,0,0, 0,0,48,24, 0,24,48,8};
```

```
TEDINFO rs_tedinfo[] = {
/* First pointer is text */
/* Second is template */
/* Third is validation */
2L, 3L, 4L, 3, 6, 2, 0x1180, 0x0, -1, 14,1,
7L, 8L, 9L, 3, 6, 2, 0x2072, 0x0, -3, 11,1,
11L, 12L, 13L, 3, 6, 0, 0x1180, 0x0, -1, 1,15,
14L, 15L, 16L, 3, 6, 1, 0x1173, 0x0, 0, 1,17};
```

```
OBJECT rs_object[] = {
/* Pointers are to: */
-1, 1, 3, G_BOX, NONE, OUTLINED, 0x21100L, 0,0, 18,12,
/* rs_strings */
2, -1, -1, G_STRING, NONE, NORMAL, 0x0L, 3,1, 12,1,
/* rs_strings */
3, -1, -1, G_BUTTON, 0x7, NORMAL, 0x1L, 5,9, 8,1,
0, 4, 4, G_BOX, NONE, NORMAL, 0xFF1172L, 3,3, 12,5,
/* rs_bitblk */
3, -1, -1, G_IMAGE, LASTOB, NORMAL, 0x0L, 3,1, 6,3,
-1, 1, 6, G_BOX, NONE, OUTLINED, 0x21100L, 0,0, 23,12,
/* rs_tedinfo */
2, -1, -1, G_TEXT, NONE, NORMAL, 0x0L, 0,1, 23,1,
6, 3, 5, G_IBOX, NONE, NORMAL, 0x1100L, 6,3, 11,5,
/* rs_strings */
4, -1, -1, G_BUTTON, 0x11, NORMAL, 0x5L, 0,0, 11,1,
/* rs_strings */
5, -1, -1, G_BUTTON, 0x11, NORMAL, 0x6L, 0,2, 11,1,
2, -1, -1, G_BOXCHAR, 0x11, NORMAL, 0x43FF1400L, 0,4, 11,1,
/* rs_tedinfo */
0, -1, -1, G_BOXTEXT, 0x27, NORMAL, 0x1L, 5,9, 13,1,
-1, 1, 3, G_BOX, NONE, OUTLINED, 0x21100L, 0,0, 32,11,
/* rs_iconblk */
2, -1, -1, G_ICON, NONE, NORMAL, 0x0L, 4,1, 6,4,
/* rs_tedinfo */
0, 4, 4, G_FBOXTEXT, 0xE, NORMAL, 0x3L, 3,5, 25,4,
/* rs_iconblk */
3, -1, -1, G_ICON, LASTOB, NORMAL, 0x1L, 1,0, 6,4};
```

/* Points to start of trees in */

```
LONG rs_trindex[] = {
0L, /* rs_object */
5L,
12L};
```

Temporary structure used by RSCREATE when setting up image pointers. */

```
struct foobar {
WORD dummy;
WORD *image;
} rs_imdope[] = {
```

```
0, &IMAG0[0],
0, &IMAG1[0],
0, &IMAG2[0],
0, &IMAG3[0],
0, &IMAG4[0];
```

/* Counts of structures defined */

```
#define NUM_STRINGS 18
#define NUM_FRSTR 0
#define NUM_IMAGES 5
#define NUM_BB 1
#define NUM_FRIMG 0
#define NUM_IB 2
#define NUM_TI 4
#define NUM_OBS 17
#define NUM_TREE 3
```

BYTE pname[] = "DEMO.RSC";

>> Title change utility <<

```
VOID
set_text(tree, obj, str)
LONG tree, str;
WORD obj;
{
LONG obspec;
```

```
obspec = LLGET(OB_SPEC(obj));
/* Get TEDINFO address */
LLSET(TE_PTEXT(obspec), str);
/* Set new text pointer */
LWSET(TE_TXTLEN(obspec), LSTRLEN(str));
/* Set new length */
}
```

>> Text edit code segment <<

```
LONG tree, obspec;
BYTE text[41];

/* Get TEDINFO address */
obspec = LLGET(OB_SPEC(EDITOBJ));
/* Set new text pointer */
LLSET(TE_PTEXT(obspec), ADDR(str));
/* Set max length */
LWSET(TE_TXTLEN(obspec), 41);
/* Make empty string */
text[0] = '\0';
```

>> Sample 68K only source code <<

```
VOID
set_text(tree, obj, str)
OBJECT *tree;
WORD obj;
BYTE *str;
{
TEDINFO *obspec;
```

obspec = (TEDINFO *) (tree + obj)->ob_spec;

/* Get TEDINFO address */

/* Set new text pointer */

obspec->te_ptext = str;

/* Set new length */

obspec->te_txtlen = strlen(str);

}

>> Symbol definitions <<

/* Window parts */

#define NAME 0x0001

#define CLOSER 0x0002

#define FULLER 0x0004

#define MOVER 0x0008

#define INFO 0x0010

#define SIZER 0x0020

#define UPARROW 0x0040

#define DNARROW 0x0080

#define VSLIDE 0x0100

#define LFARROW 0x0200

#define RTARROW 0x0400

#define HSLIDE 0x0800

/* wind_get/set parameters */

#define WF_KIND 1

#define WF_NAME 2

#define WF_INFO 3

#define WF_WXYWH 4

#define WF_CXYWH 5

#define WF_PXYWH 6

#define WF_FXYWH 7

#define WF_HSLIDE 8

#define WF_VSLIDE 9

#define WF_TOP 10

#define WF_FIRSTXYWH 11

#define WF_NEXTXYWH 12

#define WF_NEWDESK 14

/* window messages */

#define WM_REDRAW 20

#define WM_TOPPED 21

#define WM_CLOSED 22

#define WM_FULLED 23

#define WM_ARROWED 24

#define WM_HSLID 25

#define WM_VSLID 26

#define WM_SIZED 27

#define WM_MOVED 28

#define WM_NEWTOP 29

/* arrow messages */

#define WA_UPPAGE 0

#define WA_DNPAGE 1

#define WA_UPLINE 2

#define WA_DNLINE 3

#define WA_LFPAGE 4

#define WA_RTPAGE 5

#define WA_LFLINE 6

#define WA_RTLINE 7



```

/* Redraw definitions */
#define R_TREE 0
#define ROOT 0
#define MAX_DEPTH 8
/* update flags */
#define END_UPDATE 0
#define BEG_UPDATE 1
#define END_MCTRL 2
#define BEG_MCTRL 3
/* Mouse state changes */
#define M_OFF 256
#define M_ON 257
/* Object flags */
#define NONE 0x0
#define SELECTABLE 0x1
#define DEFAULT 0x2
#define EXIT 0x4
#define EDITABLE 0x8
#define RBUTTON 0x10
/* Object states */
#define SELECTED 0x1
#define CROSSED 0x2
#define CHECKED 0x4
#define DISABLED 0x8
#define OUTLINED 0x10
#define SHADOWED 0x20

#define G_BOX 20
#define G_TEXT 21
#define G_BOXTEXT 22
#define G_IMAGE 23
#define G_IBOX 25
#define G_BUTTON 26
#define G_BOXCHAR 27
#define G_STRING 28
#define G_FTEXT 29
#define G_TITLE 32
/* Data structures */
typedef struct grect
{
    int g_x;
    int g_y;
    int g_w;
    int g_h;
} GRECT;

typedef struct object
{
    int ob_next;
/* -> object's next sibling */
    int ob_head;
/* -> head of object's children */
    int ob_tail;
/* -> tail of object's children */
    unsigned int ob_type;
/* type of object- BOX, CHAR,... */
    unsigned int ob_flags;

/* flags */
    unsigned int ob_state;
/* state- SELECTED, OPEN, ... */
    long ob_spec;
/* "out"- -> anything else */
    int ob_x;
/* upper left corner of object */
    int ob_y;
/* upper left corner of object */
    int ob_width;
/* width of obj */
    int ob_height;
/* height of obj */
    } OBJECT;

typedef struct text_edinfo
{
    long te_ptext;
/* ptr to text (must be 1st) */
    long te_ptmpl;
/* ptr to template */
    long te_pvalid;
/* ptr to validation chrs. */
    int te_font;
/* font */
    int te_junk1;
/* junk word */
    int te_just;
/* justification- left, right */
    int te_color;
/* color information word */
    int te_junk2;
/* junk word */
    int te_thickness;
/* border thickness */
    int te_txtlen;
/* length of template string */
    } TEDINFO;

/* "Portable" data definitions */
#define OB_NEXT(x) (tree + (x) * sizeof(OBJECT) + 0)
#define OB_HEAD(x) (tree + (x) * sizeof(OBJECT) + 2)
#define OB_TAIL(x) (tree + (x) * sizeof(OBJECT) + 4)
#define OB_TYPE(x) (tree + (x) * sizeof(OBJECT) + 6)
#define OB_FLAGS(x) (tree + (x) * sizeof(OBJECT) + 8)
#define OB_STATE(x) (tree + (x) * sizeof(OBJECT) + 10)
#define OB_SPEC(x) (tree + (x) * sizeof(OBJECT) + 12)
#define OB_X(x) (tree + (x) * sizeof(OBJECT) + 16)
#define OB_Y(x) (tree + (x) * sizeof(OBJECT) + 18)
#define OB_WIDTH(x) (tree + (x) * sizeof(OBJECT) + 20)
#define OB_HEIGHT(x) (tree + (x) * sizeof(OBJECT) + 22)

#define TE_PTEXT(x) (x)
#define TE_TXTLEN(x) (x + 24)

```

Classified

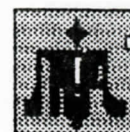
FOR SALE -- FOR SALE -- FOR SALE

One ATR-8000 CP/M add on for the Atari 800, XL and XE computers. The ATR-8000 will allow you to add standard disk drives to your Atari computer. It will run single, double or quad density, single or double sided drives in both the Atari and CP/M modes. It has an RS232 port for modems, printers or a terminal. A parallel printer port that has a 48K print buffer when in the Atari mode is also on this unit. The ATR also runs standard CP/M programs and has too many features in this mode to mention here.

System consists of the ATR-8000, two SSDD disk drives with case and power supply, 9" USI amber monitor and either the 80 col. cartridge or an 80 col chip for an 800XL.

Price -- \$500

If interested call Gary Nolan at 353-9716 or see me at the next meeting and we'll discuss it.



From Page 4 AU MAN, THAT'S NOT EVEN FAIR (TELL ME MORE, TELL ME MORE)

PSSST, move in a little closer we don't want everyone to hear this.

You say you like Donkey Kong Jr. but get too frustrated after playing for a while? Listen close. After you start the game, pause it (press the space bar, son) hold down the "shift" key and type the word BOOGA. Now un-pause the game (space bar again) and press the "S" key to change screens or the "K" key to make yourself immune to those pesky snappers, birds and other nasties. Trust me. Would I steer you wrong. And if you're nice to me, I might tell you what I know about Ghostbusters and Preppie. I said NICE, now.

ENOUGH IS ENOUGH

That's about it for this month. Remember the elections next month and the MIDI demo in June.

See you on the 17th.....

Did I mention that someone was selling an ATR system?? Look for the ad.

Non - Fattening Fund Raiser

We found a fund raiser!! This one requires no selling, and is non Fattening. This Fund raiser involves mainly the women-members or wives of members but the men are always welcome to join in the fun.

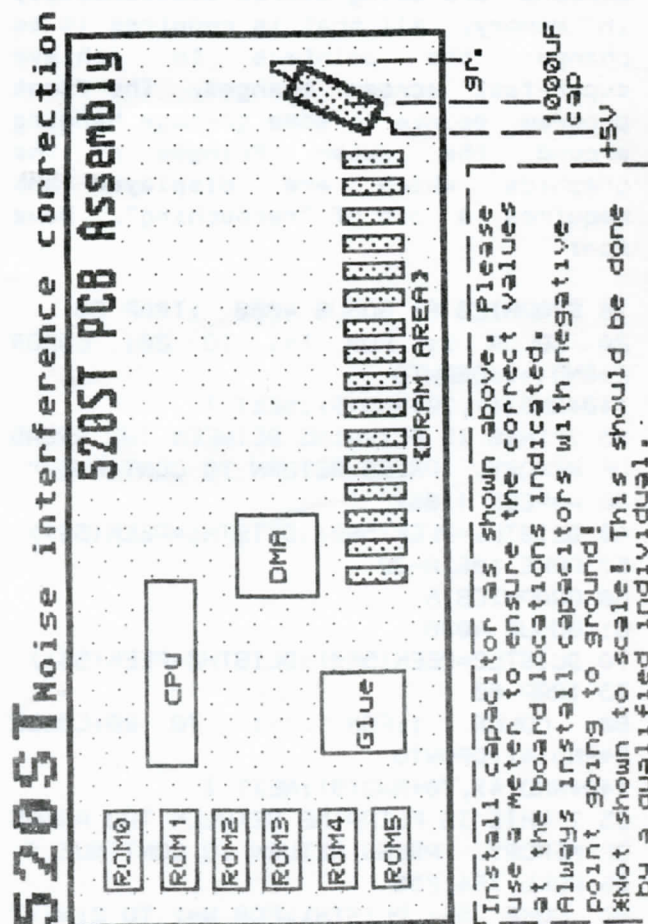
Simply, a number of women hold TUPPERWARE parties. The hostess still receive their Thank-you gift and MILATARI receives 10 - 15 % of the total sales.

The more parties the better but any parties will benefit the club. You don't have to go door to door and TUPPERWARE is non-fattening.

Male members, even if you are not interested show this article to your spouse (or girlfriend) and ask them to have a party to support our club. They can hold parties even if they themselves are not members.

For more information I will be at the May meeting. Or for more information call me at 462-8178 I will be happy to answer any questions regarding this offer. Please make any calls before 10 P.M.. Thank You.

Heidi Tupper



Reprinted from
Toronto Phenix



Basic Column
by A.L.D.

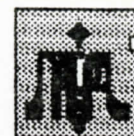
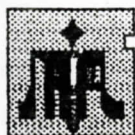
reprinted from THE SOUTHWEST VALLEY ATARI CONNECTION

Buried away in my collection of disks I found a couple of novel programs. From whence they came or where they may lead...I cannot even hazard a guess, but they illustrate some very nice features of semi-relocatable memory which the ATARI has and no other 6502 has. The screen display in the ATARI is "relocatable", meaning that you can put one display screen display in one location in memory and another screen in another location in memory (called pages on the APPLE). These two programs illustrate this feature. Pointers are sorted in 560 and 561 telling the computer where each screen is stored in memory. Since multiple screens are being stored simultaneously in memory, all that is required is to change the pointers to achieve super-fast screen changes. The first program points to some garbage hanging around the outer fringes of the graphics which are displayed and requires a bit of "retouching". Have fun:

```
10 GRAPHICS 6: GOSUB 4000 :TRAP 20
20 COLOR 1: FOR I=1 TO 20: COLOR
2*RND(4):DRAWTO
140*RND(4),70*RND(9):NEXT I
25 ?"THIS IS FLIPPING BETWEEN TWO AREAS
OF MEMORY. PRESS RETURN TO CONTINUE."
30 A=PEEK(106)
40 DLIST1=PEEK(560):DLISTH1=PEEK(561)
50 POKE 106,A-32
60 GRAPHICS 6
61 GOSUB 4000
70 DLIST2=PEEK(560):DLISTH2=PEEK(561)
75 TRAP 80
80 COLOR 1:FOR I=1 TO 20:COLOR
2*RND(4):DRAWTO
140*RND(4),70*RND(9):NEXT I
85 ?"THIS IS FLIPPING BETWEEN TWO AREAS
OF MEMORY. PRESS RETURN TO CONTINUE."
86 POKE 764,255
90 POKE 561,DLISTH1:FOR W=1 TO 2:NEXT
W:POKE 561,DLISTH2:IF PEEK(764)=12 THEN
110
100 GOTO 90
110 POKE 106,A:RUN "D:NEXT1"
4000 X=PEEK(16):IF X=129 THEN 4020
4010 POKE 16,X-128:POKE 53774,X-128
4020 RETURN
```

- Phoenix, AZ

```
5 TRAP 10
10 GRAPHICS 0
20 POSITION 4,10:?"THIS IS PAGE
1.":POSITION 2,20:?"PRESS 1,2,3 OR 4
FOR THAT PAGE."
25 ?"PRESS RETURN TO CONTINUE."
30 A=PEEK(106)
40 DLL1=PEEK(560):DLH1=PEEK(561)
45 REM *****
50 POKE 106,A-B
60 GRAPHICS 0
70 POSITION 10,10:?"THIS IS PAGE
2.":POSITION 2,20:?"PRESS 1,2,3 OR 4
FOR THAT PAGE."
75 ?"PRESS RETURN TO CONTINUE."
80 DLL2=PEEK(560):DLH2=PEEK(561)
85 REM *****
90 POKE 106,A-16
100 GRAPHICS 0
110 POSITION 15,10:?"THIS IS PAGE
3.":POSITION 2,20:?"PRESS 1,2,3 OR 4
FOR THAT PAGE."
120 DLL3=PEEK(560):DLH3=PEEK(561)
125 REM *****
130 POKE 106,A-24
140 GRAPHICS 0
150 POSITION 20,10:?"THIS IS PAGE
4.":POSITION 2,20:?"PRESS 1,2,3 OR 4
FOR THAT PAGE."
155 ?"PRESS RETURN TO CONTINUE."
160 DLL4=PEEK(560):DLH4=PEEK(561)
165 REM *****
170 CH=PEEK(560):DLH4=PEEK(561)
165 REM *****
170 CH=PEEK(764)
180 IF CH=31 THEN 190
181 IF CH=30 THEN 200
182 IF CH=26 THEN 210
183 IF CH=24 THEN 220
184 IF CH=12 THEN 230
185 GOTO 170
190 POKE 560,DLL1:POKE 562,DLH1:GOTO
170
200 POKE 560,DLL2:POKE 561,DLH2:GOTO170
210 POKE 560,DLL3:POKE561,DLH3:GOTO 170
220 POKE 560,DLL4:POKE 561,DLH4:GOTO
170
230 POKE 106,A:RUN "D:NEXT2"
4000 X=PEEK(16):IF X=128 THEN 4020
4010 POKE 16,X-128:POKE 53774,X-128
4020 RETURN
```

204K II SOLUTION

"TAKE ALL", "S", "S", "S", "SW", "S", "LIGHT LAMP", "SE",
 "IN", "TAKE TEAPOT", "OUT", "N", "NE", "TAKE WATER",
 "S", "SE", "S", "S", "W", "SE" UNTIL YOU GET TO THE
 RIDDLE ROOM, ANSWER TO RIDDLE: "A WELL", "E", "TAKE
 PEARLS", "E", "GET IN BUCKET", "POUR WATER", "GET OUT",
 "E", "TAKE RED CAKE, BLUE CAKE, AND GREEN CAKE", "EAT
 GREEN CAKE", "E", "THROW RED CAKE IN POOL", "TAKE CANDY",
 "W", "EAT BLUE CAKE", "NW", "TELL ROBOT", "E", "E",
 "TELL ROBOT", "PUSH TRIANGLE", "TELL ROBOT", "S", "S",
 "TAKE SPHERE", "TELL ROBOT", "LIFT
 CAGE", "TAKE SPHERE", "N", "W", "S", "W", "GET IN
 BUCKET", "TAKE WATER", "GET OUT", "DROP TEAPOT", "W",
 "W", "NW", "OPEN BOX", "TAKE VIOLIN", "E", "N", "N",
 "IN", "DROP ALL BUT LAMP AND SWORD", "TAKE MAT AND
 OPENER", "OUT", "S", "S", "N", "TAKE CLAY", "N",
 "N", "U", "PUT MAT UNDER DOOR", "OPEN LID", "PUT OPENER
 IN HOLE", "TAKE HAT", "TAKE KEY", "TAKE OPENER", "UNLOCK
 DOOR WITH KEY", "OPEN DOOR", "W", "TAKE SPHERE", "S",
 "D", "S", "S", "E", "N", "N", "IN", "DROP ALL BUT
 LAMP, SWORD, AND BRICK", "TAKE NEWSPAPER AND MATCHES",
 "OUT", "S", "S", "E", "SW", "TAKE STRING", "N", "D",
 "E", "N", "N", "ATTACK DRAGON", "S", "ATTACK DRAGON",
 "S", "ATTACK DRAGON", "W", "W", "TAKE RUBY", "S", "GET
 IN BASKET", "OPEN RECEPTACLE", "PUT PAPER IN
 RECEPTACLE", "LIGHT MATCH", "LIGHT PAPER WITH MATCH",
 WAIT, WAIT, "W", "TIE WIRE TO HOOK", "GET OUT", "TAKE
 COIN", "S", "OPEN PURPLE BOOK", "TAKE STAMP", "N", "GET
 IN", "UNTIE WIRE", WAIT, WAIT, WAIT, WAIT, "W", "TIE
 WIRE TO HOOK", "GET OUT", "S", "PUT STRING IN BRICK",
 "PUT BRICK IN HOLE", "LIGHT MATCH", "LIGHT STRING WITH
 MATCH", "N", "S", "TAKE CROWN", "N", "GET IN", "UNTIE
 WIRE", "CLOSE RECEPTACLE", WAIT, (UNTIL BASKET REACHES
 VOLCANO BOTTOM) "GET OUT", "N", "E", "E", "SE", "E", "N",
 "N", "IN", "DROP ALL BUT LAMP", "OUT", "S", "W", "SW",
 "N", "N", "N", "W", "N", "N", "TAKE CHEST", "KISS
 PRINCESS", "S", "S", "S", "SE", "E", "N", "N", "WAIT"
 FOR PRINCESS, "OPEN CHEST", "AGAIN", "TAKE DRAGON",
 "DROP CHEST AND ROSE", "TAKE CANDY, RED SPHERE, BLUE
 SPHERE, AND PEARLS", "OUT", "S", "S", "W", "S", "S",
 "D", "S", "TAKE CLUE", "SE", "NE", "NW", "SW", "N", "U",
 "N", "N", "SW", "SW", "FEED LIZARD CANDY", "UNLOCK DOOR
 WITH GOLD KEY", "OPEN DOOR", "S", "W", "DROP ALL BUT
 LAMP", "E", "N", "N", "NE", "N", "R", "N", "W", "N",
 "W", "W", "NE", "E", "S", "TAKE PORTRAIT", "N", "ENTER
 LIGHT", "ENTER S WALL", "ENTER LIGHT", "TAKE BILLS",
 "KILL", "W", "W", "W", "TAKE LAMP", "S", "S", "S", "SW",
 "S", "SE", "IN", "TAKE ALL", "OUT", "S", "S", "W", "SW",
 "SW", "S", "E", "DROP ALL BUT LAMP", "TAKE CLUB", "W",
 "THROW CLUB AT GLASS", "TAKE WHITE SPHERE", "E", "PUT
 WHITE SPHERE ON DIAMOND STAND", "PUT RED SPHERE IN RUBY
 STAND", "PUT BLUE SPHERE ON SAPPHIRE STAND", "TAKE BLACK
 SPHERE", "S", "PUT SPHERE IN CIRCLE", "N", "TAKE RUBY,

COIN, STAMP, AND VIOLIN", "S", "GIVE ALL BUT LAMP TO
 DEMON", "N", "TAKE ALL", "S", "GIVE ALL BUT LAMP TO
 DEMON", "TELL DEMON", "GIVE ME HAND", "TAKE HAND", "N",
 "E", "N", "N", "NE", "S", "POINT HAND AT MENHIR", "SAY
 FLOAT", "SW", "TAKE COLLAR", "NE", "S", "D", "D", "PUT
 COLLAR ON DOG", "E", "OPEN DOOR", "S", "TURN OFF LAMP",
 "OPEN SECRET DOOR", "S".

YOU WIN! HURRAY!!

Reprinted from
 ORNJ/UCF

They Laughed at my Program so...
 by The Deve (ACAOC?)

Six-pack of floppies in hand, I sauntered off to last month's user group meeting. Since lots of folks like to trade programs at these meetings, I grabbed some stuff that I had written, eager to share it with the group.

The trading group was on the far side of the room so I had to elbow my way through a gaggle of arcade game nerds then slowly circle a pack of word processor junkies debating the pros and cons of 80-column Letter Perfect. I was in a bad mood already, having just read in ANALOG that every piece of computer hardware I own will soon be obsolete.

When I finally reached the trading group over by the door, I booted up my first program for the crowd. "What's this garbage!?" scoffed a skinny 14-year-old kid.

"VisiLaundryList," I calmly exclaimed, restraining myself from wringing his neck. "It sorts out your dirty socks and underwear according to the type of stain and the date you threw it under the bed." A few users suppressed outright laughter, but the guy in the T-shirt that said, "I've got a hard-disk drive!" smirked as he ripped out my disk.

"Hold it," I said, slipping in a copy of Cosby-Calc. "This one's really handy--it keeps track of the number of times you see Bill Cosby in TV commercials, indexing them by product and other keywords." That's when they all burst out laughing, at me, not with me.

"Come here, everybody!" the kid shouted, and 87 users huddled around, chortling and giggling while he scornfully ridiculed the program I'd spent three months writing. I just couldn't take it anymore. That's when I ran over to the janitor's closet, grabbed an axe and lopped off his head.

"Debug this!" I screamed, slashing savagely at the hysterical hackers until the room ran with blood. I then went back for the chainsaw

MILWAUKEE AREA ATARI USER'S GROUP AND NEWSLETTER INFORMATION

MILATARI OFFICERS

President	David Frazer	542-7242
Vice President	Carl Mielcarek	355-3539
Secretary	Currently open	
Treasurer	Steve Tupper	462-8178

MILATARI VOLUNTEERS

MILATARI West	Dennis Bogie	968-2361
Education SIG	Joe Sanders	447-1660
ATR8000 & CP/M	Joe Griesemer	673-6663
Database	Ron Friedel	354-1717
BBS Sysop	Richard Dankert	781-2338
Membership	Dave Bogie	968-2361
Newsletter	Roy Duvall	363-8231

Libraries

Cassette &		
Copyright	Lee Musial	466-9160
Disk	Dennis Wilson	476-7767
	Bill Lawrence	968-3082
Publications	Bill Feest	321-4314

Milatari BBS

300 Baud 24hrs. 414-781-5710

NEWSLETTER INFORMATION

This newsletter is written and printed by members of the Milwaukee Area ATARI User's Group [MILATARI]. Opinions expressed in this publication are those of the individual authors and do not necessarily represent or reflect the opinions of the Milwaukee Area Atari User's Group, its officers, its members or its advertisers except where noted. Articles reprinted from bulletin boards and other newsletters are presented as an information exchange, no warranty is made to their accuracy or factuality.

Your contributions of articles are always welcome. You may submit your article on ATARI compatible cassette or diskette, on typewritten form or you can arrange with the editor to upload your file via modem. You can send Graphics eight or seven plus screens stored on disk in Micropainter or Micro Illustrator formats.

Milwaukee Area Atari User's Group

MILATARI is an independent, user education group which is not affiliated with ATARI INC. The newsletter is the official publication of MILATARI and is intended for the education of its members as well as for the dissemination of information concerning ATARI computer products.

MILATARI membership is open to individuals and families who are interested in using and programming ATARI computers. The membership includes a subscription to this newsletter and access to the club libraries. The annual membership fee is \$15 for individuals or \$20 for a family.

Vendors wishing to display and/or sell items at MILATARI meetings must make prior arrangements with the club vice president. Rates are \$10 per meeting or \$90 per year payable in advance.

All material in this newsletter bearing a COPYRIGHT message may be reprinted in any form, provided that MILATARI and the author are given credit.

Other computer user groups may obtain copies of this newsletter on an exchange basis.

MILATARI ADVERTISING RATES

This newsletter will accept camera ready advertising copy from anyone supplying goods and services of interest to our membership.

Current paid members of MILATARI may place classified ads in the newsletter at no charge.

Advertising Rates

Full page	\$37.50
Half page	\$20.00
Quarter page	\$12.50
Business card	\$2.00

**MILATARI BBS 414-781-5710 24 Hours
300 Baud**

\$500

256 K 800 XL 130 XE Compatable

Yes, for \$5.00 you can purchase a high
quality circuit board that makes
converting your 800XL a snap!

Includes: Bare board, Documentation,
and software on disk*.

Additional parts Required

~~~~~  
Populated board ..... \$20.00  
Populated board with 256K RAM .. \$50.00

\*Disks on exchange basis or available at \$5.00 to cover handling & postage

**QUANTITIES LIMITED TO AVAILABLE STOCK**

**HURRY ORDER YOURS TODAY!**

**DENNIS BOGIE**

S 35 W 28674 Loon Hollow  
Waukesha, WI 53188  
or call 414-968-2361



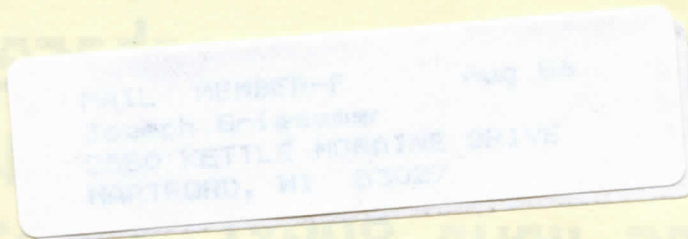


MILWAUKEE AREA ATARI USERS GROUP  
Post Office Box 19858  
West Allis, Wisconsin 53219-0858

=====

Address Correction Requested  
Forwarding Postage Guaranteed

=====



## 69 CENT DISKS ARE HERE!

High Quality, 5-1/4", Single Sided (punched for two sided use),  
Double Density, complete with hub rings, write protect labels and  
Tyvek sleeves.

|                 |         |
|-----------------|---------|
| 10 Disks . . .  | \$6.90  |
| 50 Disks . . .  | \$34.50 |
| 100 Disks . . . | \$69.00 |

Disks sold only in  
multiples of 10

Prices and availability are subject to change without notice

**COMPUTER SOFTWARE CENTER**  
9805 W. Oklahoma Ave., Milwaukee

(Two blocks East of Interstate 594)

Tues.-Fri. 12-8, Sat. 12-5

(414) 543-5123



THERE'S MORE IN STORE FOR YOU AT 98TH & OKLAHOMA